Does Partial Replication Pay Off?

FTXS'12 Workshop
June 25, 2012

Jon Stearley jrstear@sandia.gov

With Kurt Ferreira, David Robinson, Jim Laros, Kevin Pedretti,
Dorian Arnold, Patrick Bridges, Rolf Riesen

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Outline

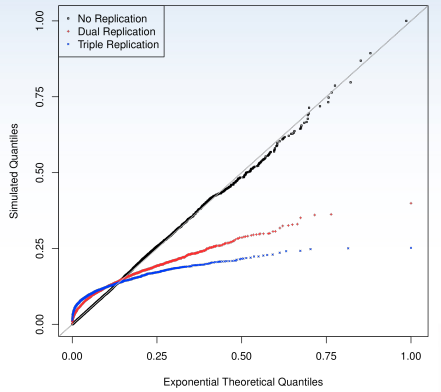
For HPC systems using process replication:

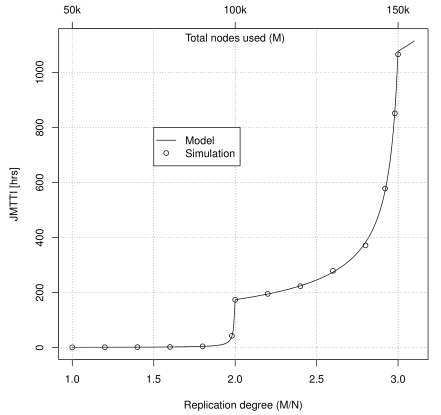
1. Time to job interrupt is NOT exponentially distributed!

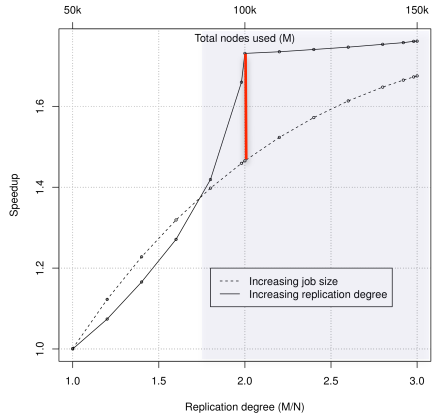
   This is fundamentally different from:
   - models of current systems (including optimal checkpointing)
   - other replication-based models (e.g. *Combining Partial Redundancy and Checkpointing for* HPC, J Elliot et al, ICDCS 2012)

2. Job mean time to interrupt (JMTTI) increases exponentially with replication degree!

3. Partial replication DOES pay off,
   but full replication degrees offer the greatest value.







Assumptions

1. Items fail independently and identically (IID).

2. Time to failure of each item is exponentially distributed. (We use a mean of 5 years.)

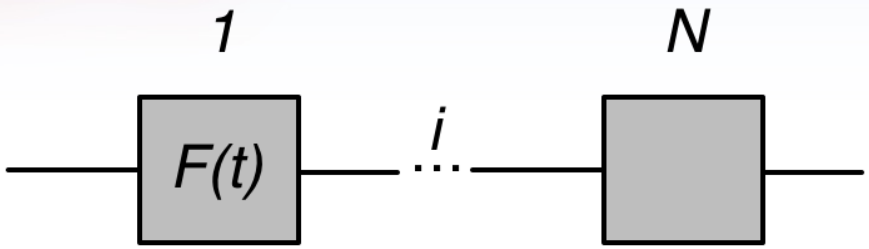
$$F(t) = 1 - e^{-t/\theta} \qquad \theta = 5yr$$

F(t) is a Cumulative Distribution Function (CDF),
and gives the probability that the item has failed **by** time t.

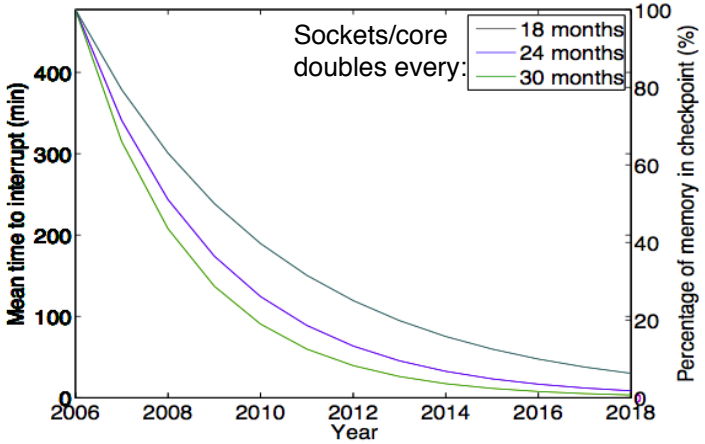
f(t)=F'(t) is a Probability Density Function (PDF),
and gives the probability that the item has failed **at** time t.

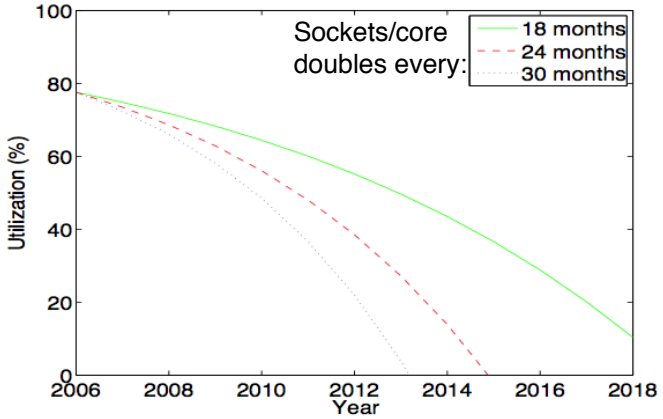
3. Items are immediately repaired (repair time is zero).

HPC System



More items, sooner interrupts:





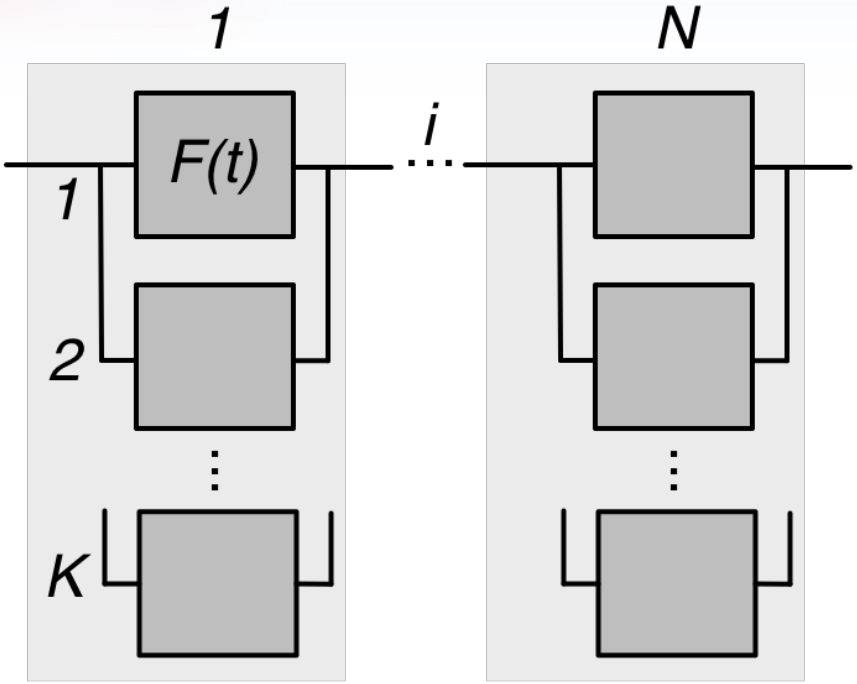
Items fail IID as *F(t)*.

Job runs on *N* ranks.

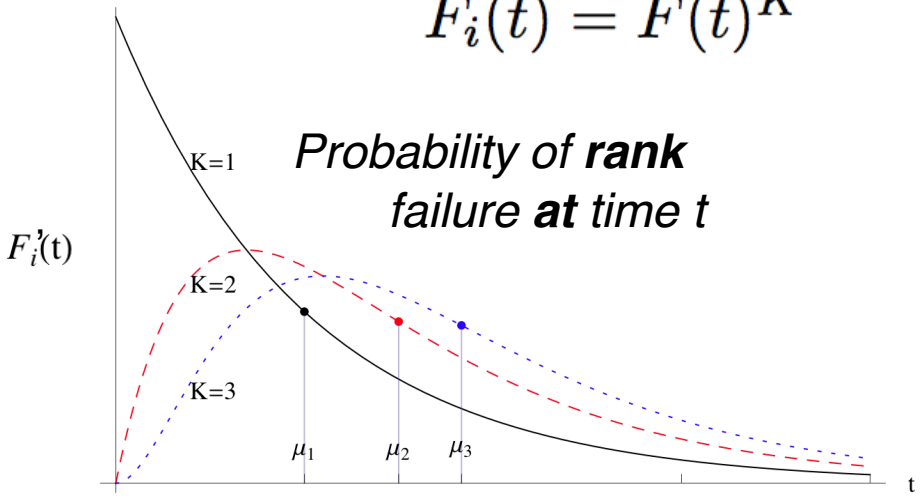
When any rank fails, the job is interrupted.

Plots from “Understanding failures in petascale computers”, Schroeder and Gibson, 2007 SciDAC Journal of Physics.

HPC System
with Uniform Process Replication



$$F_i(t) = F(t)^K$$



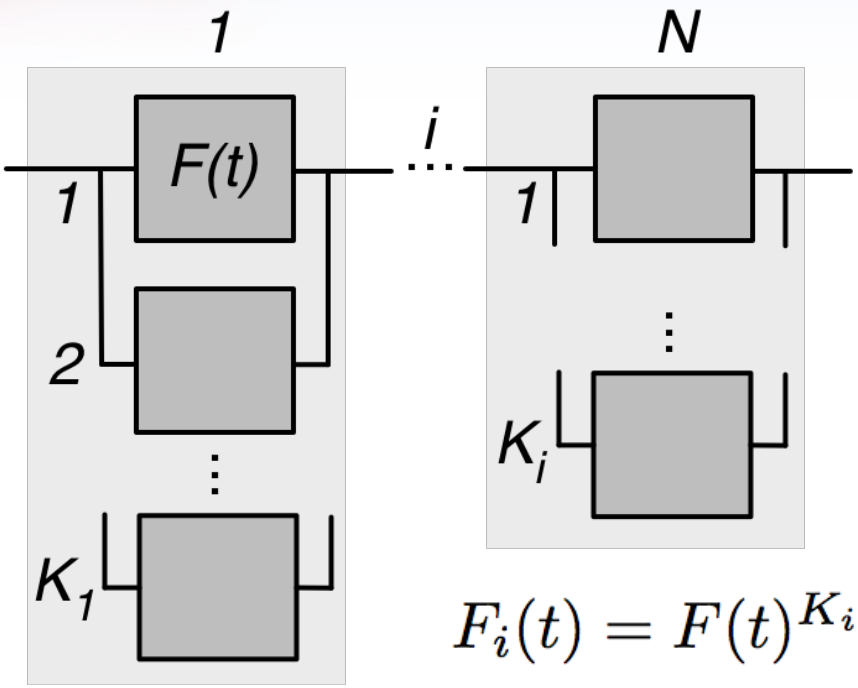
Items fail IID as *F(t)*.

Job runs on *N* ranks.

When all *K* replica items in a rank fail, the rank fails.

When any rank fails, the job is interrupted.

General Formulation



$$F_i(t) = F(t)^{K_i}$$

$$F_j(t) = 1 - R_j(t)$$

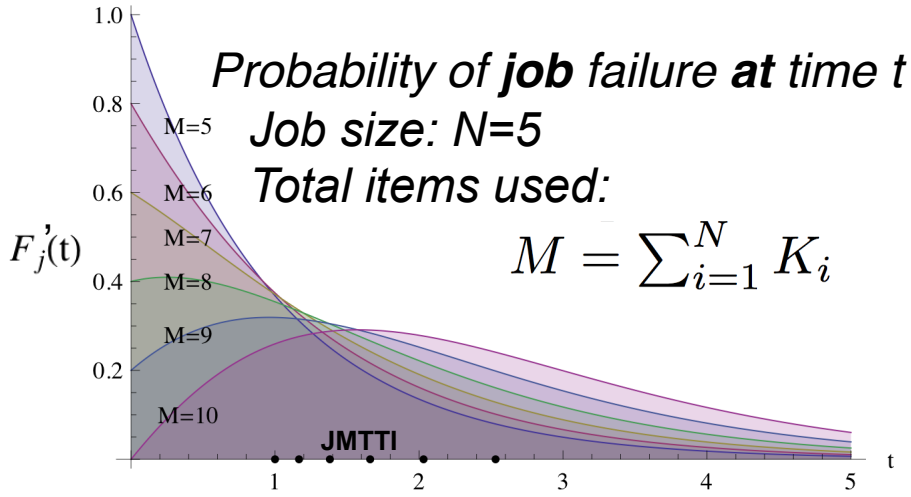
$$R_j(t) = \prod_{i=1}^{N} (1 - F(t)^{K_i})$$

Items fail IID as *F(t)*.

Job *j* runs on *N* ranks.

When all K_i replica items in rank *i* fail, the rank fails.

When any rank fails, the job is interrupted. Job mean time to interrupt is JMTTI.



*Probability of **job** failure **at** time t*

Job size: N=5

Total items used:

$$M = \sum_{i=1}^{N} K_i$$

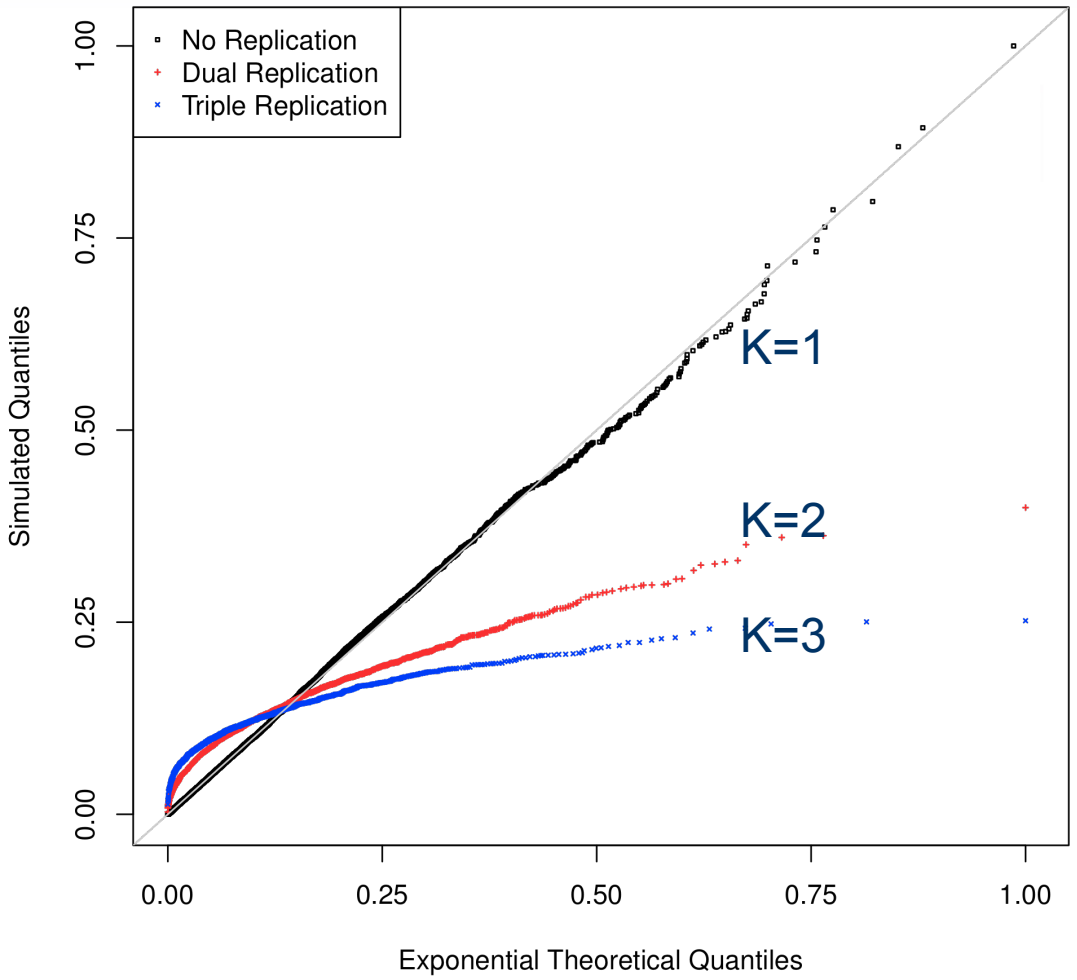
Job Time To Interrupt (JTTI) is NOT exponentially distributed!

Q-Q plots are a standard way of comparing distributions.

The median value is the 50% quantile.

Quantiles from two distributions are compared; if X=Y then the distributions match.

JTTI is clearly not exponential for K>1



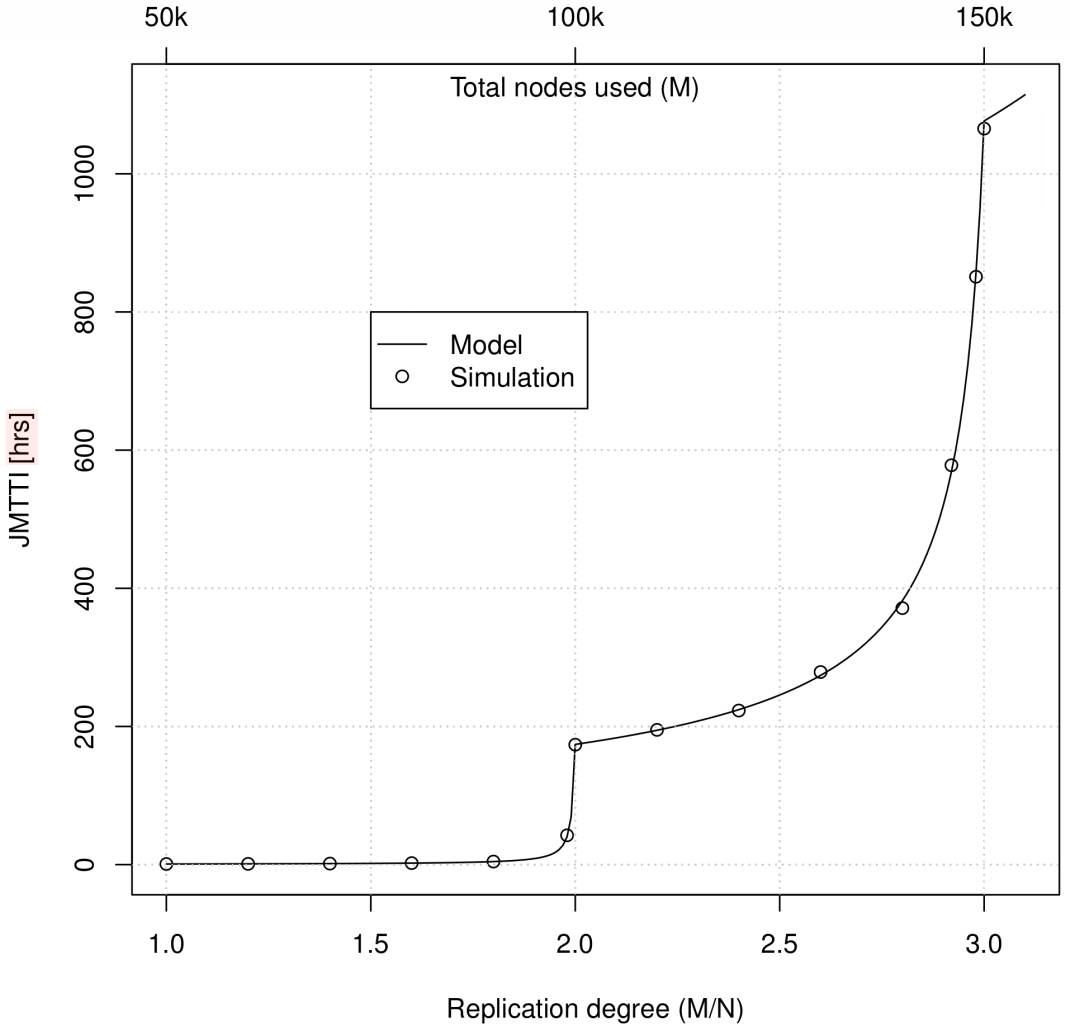
Job Mean Time To Interrupt (JMTTI) increases exponentially!

Exponential across partial replication regions.

Exponential at successive full replication degrees.

Plateaus at full replication degrees not previously studied.

M/N	JMTTI [minutes]
1.0	52
1.2	65
1.4	88
1.6	131
1.8	266
2.0	10,416



$$F(t) = 1 - e^{-t/\theta}$$

$$R_j(t) = \prod_{i=1}^{N} (1 - F(t)^{K_i})$$

$$JMTTI = \int_0^{\infty} R_j(t)dt$$

Total Wallclock Time of a job using process replication and checkpointing

Daly's Model for wallclock time $T_w = JMTTI * e^{R/JMTTI}\left(e^{(\tau+\delta)/JMTTI} - 1\right)\frac{T_s}{\tau}$

and optimal checkpoint interval $\tau_{opt} = A * \sqrt{2 * JMTTI * \delta} - \delta$

$$A = 1 + \frac{\delta}{18 * JMTTI} + \sqrt{\frac{\delta}{18 * JMTTI}}$$

Assumes that job interrupts are exponentially distributed - which we have shown is not true for replication systems.

Replication-based models of these is left for future work…

However we take an initial look via simulation, setting checkpoint intervals to τ_{opt} above, using JMTTI below:

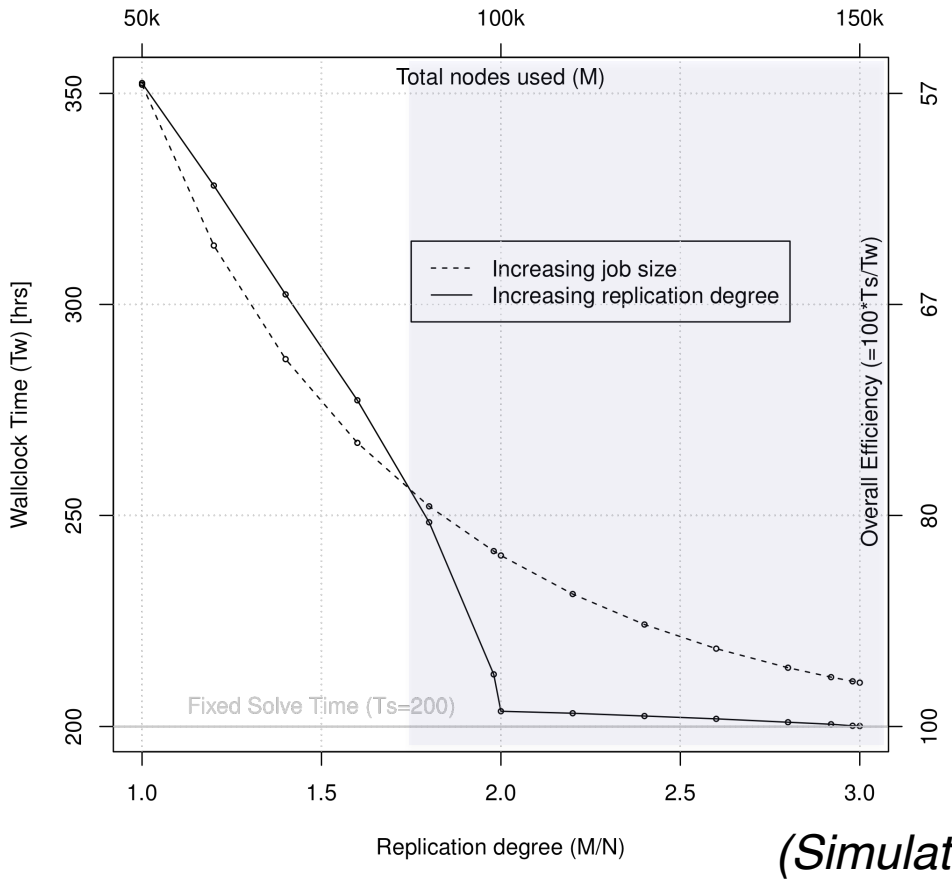
$$F(t) = 1 - e^{-t/\theta}$$

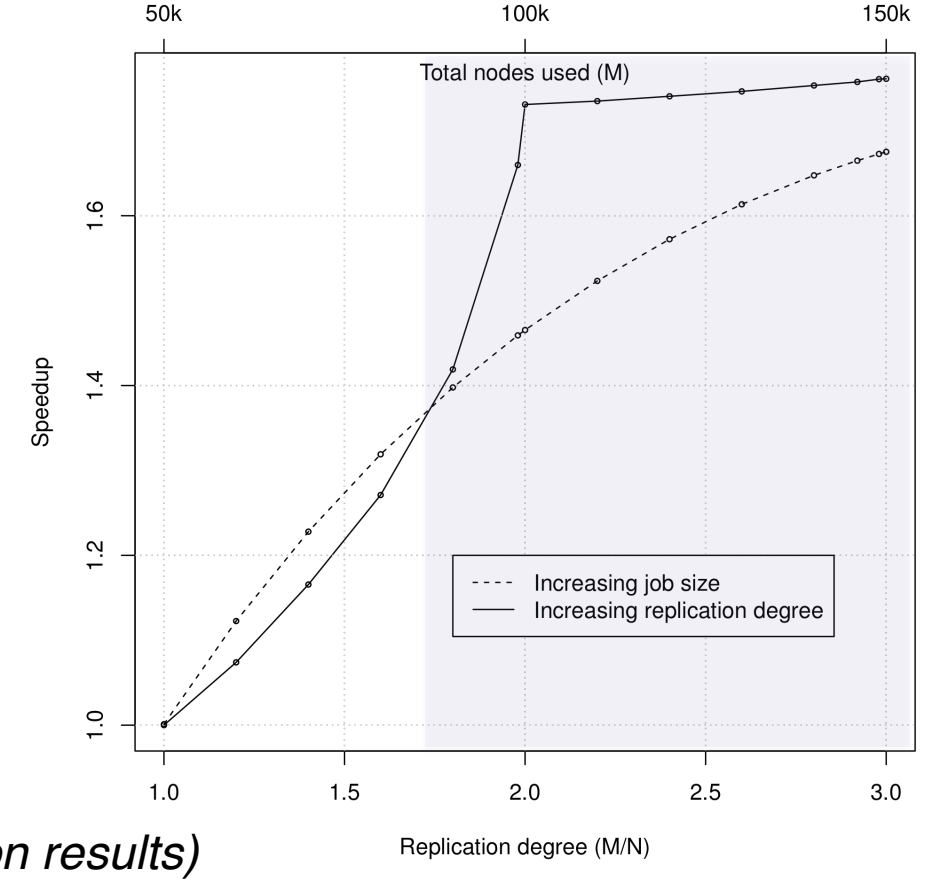
$$R_j(t) = \prod_{i=1}^{N}(1 - F(t)^{K_i})$$

$$JMTTI = \int_0^\infty R_j(t)dt$$

Partial Replication Pays Off

In the shaded region, replication yields better speedup than perfect strong scaling (using items to increase replication has paid off more than using them to increase job size).



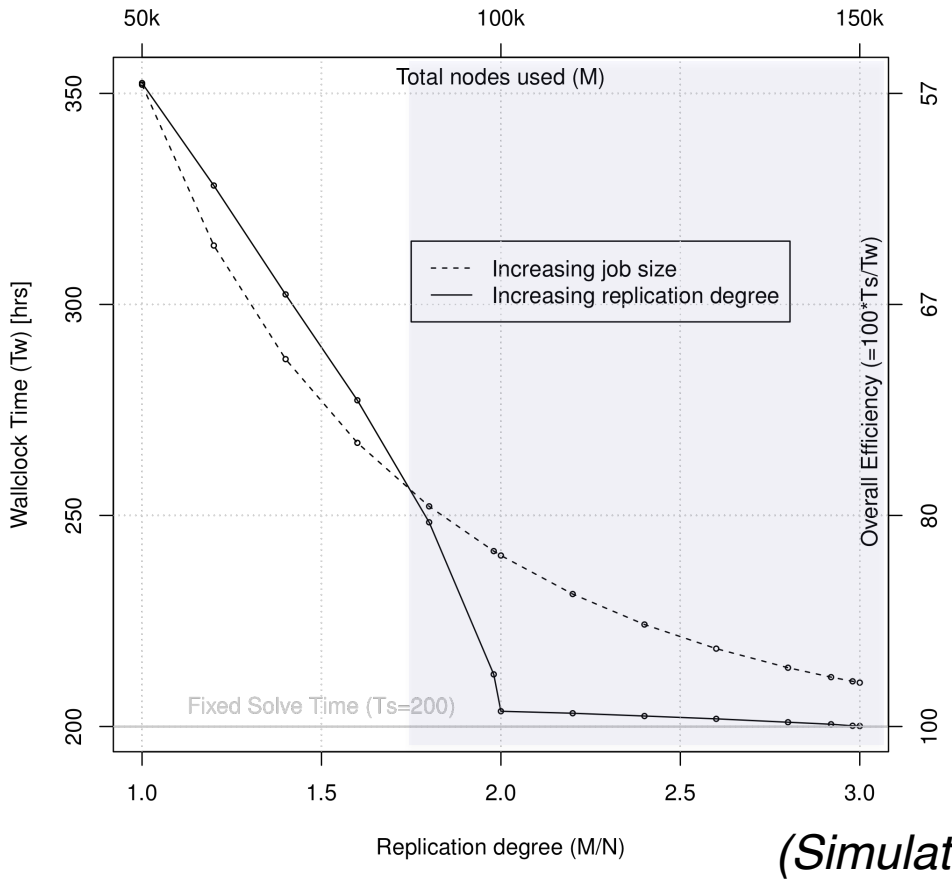


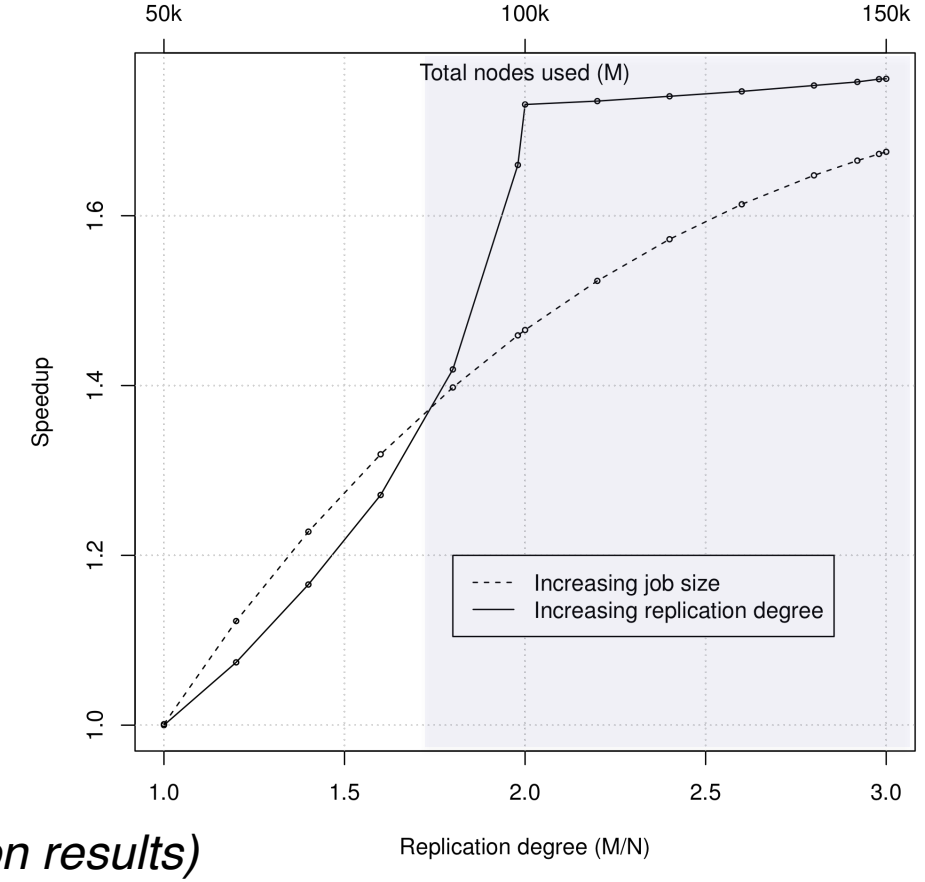
(Simulation results)

Partial Replication Pays Off

M/N	JMTTI [minutes]	Wasted jobs
1.0	52	43%
1.2	65	39%
1.4	88	34%
1.6	131	28%
1.8	266	20%
2.0	10,416	.05%

AS JMTTI INCREASES WITH REPLICATION DEGREE M/N, FEWER JOBS ARE INTERRUPTED BEFORE THEY CAN WRITE THEIR FIRST CHECKPOINT. WE CALL SUCH JOBS "WASTED" BECAUSE THEY MAKE NO FORWARD PROGRESS.

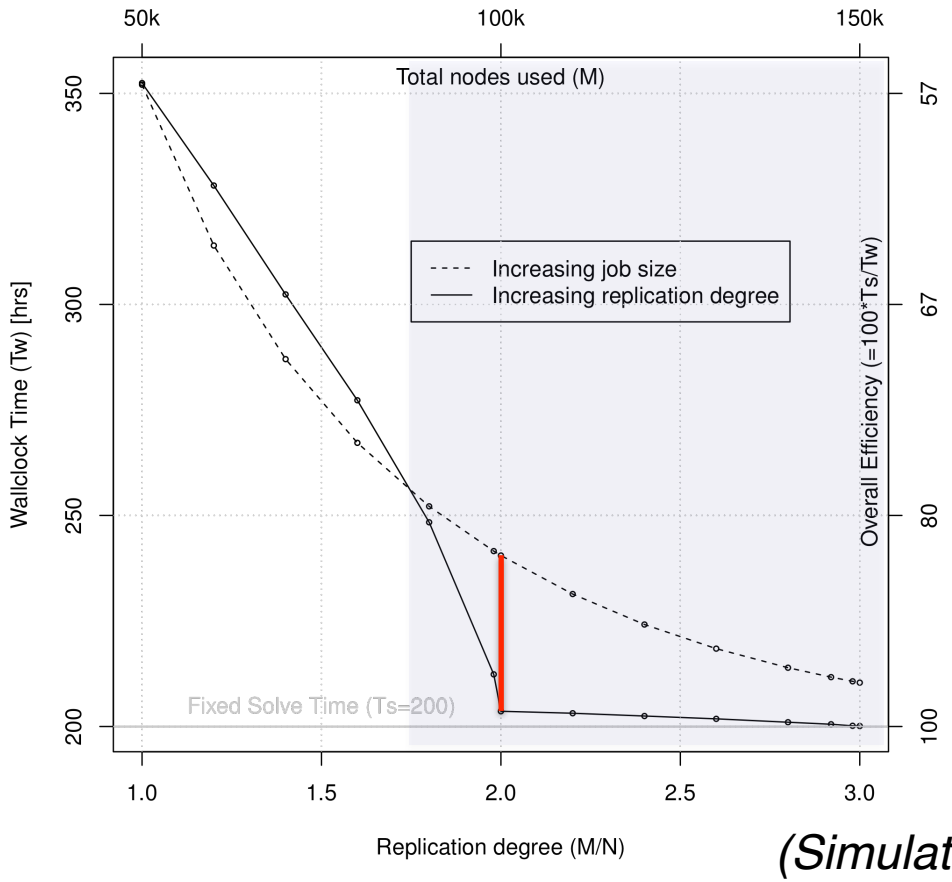


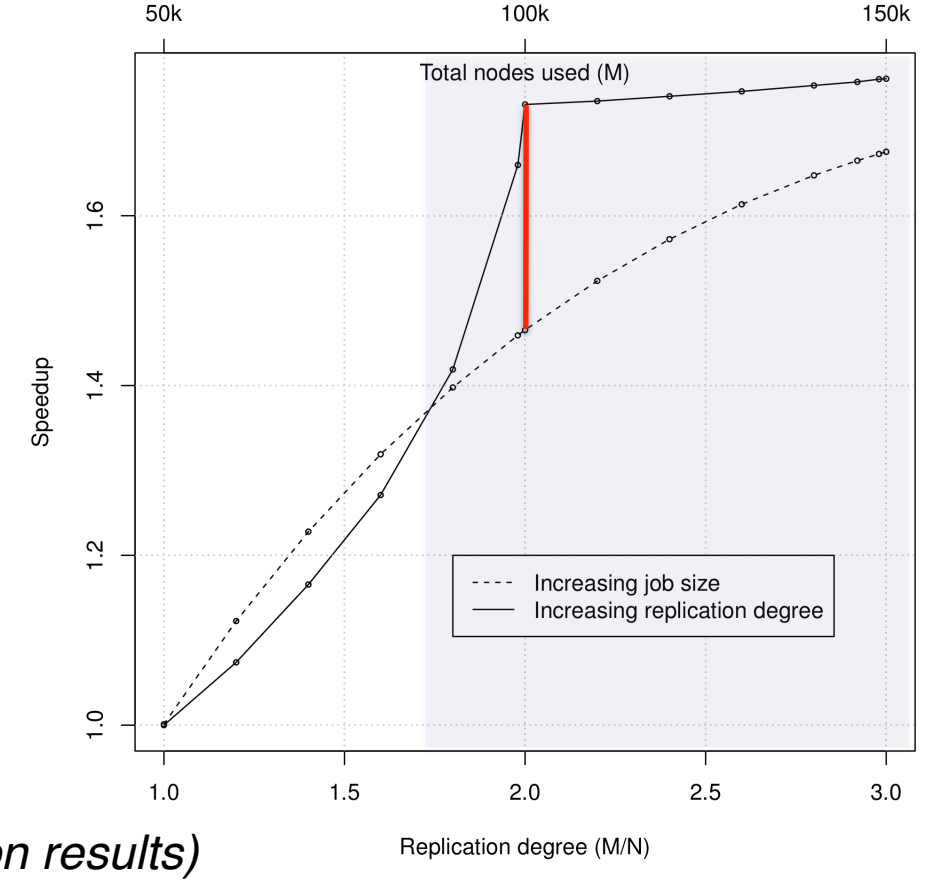


(Simulation results)

Partial Replication Pays Off, but full replication offers the best value!

The ratio of speedup to total number of items used is greatest at full replication levels (just like JMTTI plateaus).





(Simulation results)

Conclusions

Do not assume that process replication is not the best way to accomplish HPC resilience! It also enables silent error detection/correction, and how about replacing failed replica items while the job is still running?

Time to job interrupts are NOT exponentially distributed!

- even when item failures ARE (and they are probably not anyway).
- Revised wallclock and optimal checkpoint interval solutions are needed.

Job mean time to interrupt (JMTTI) increases exponentially with replication degree!

- A simple way to estimate JMTTI would be nice (e.g. Bougeret, Casanova, Robert, et al)

Partial replication DOES pay off, but full replication degrees offer the greatest value.

- When should replication be used? Eg what runtime or reliability goal, for applications with what scaling characteristics, including what replication overhead?